Centaur Intelligence: GPT-4 and Mathematica Show Maximum Topological Compactness of the Genetic Code Using Isotropic Axes Ordered GACU

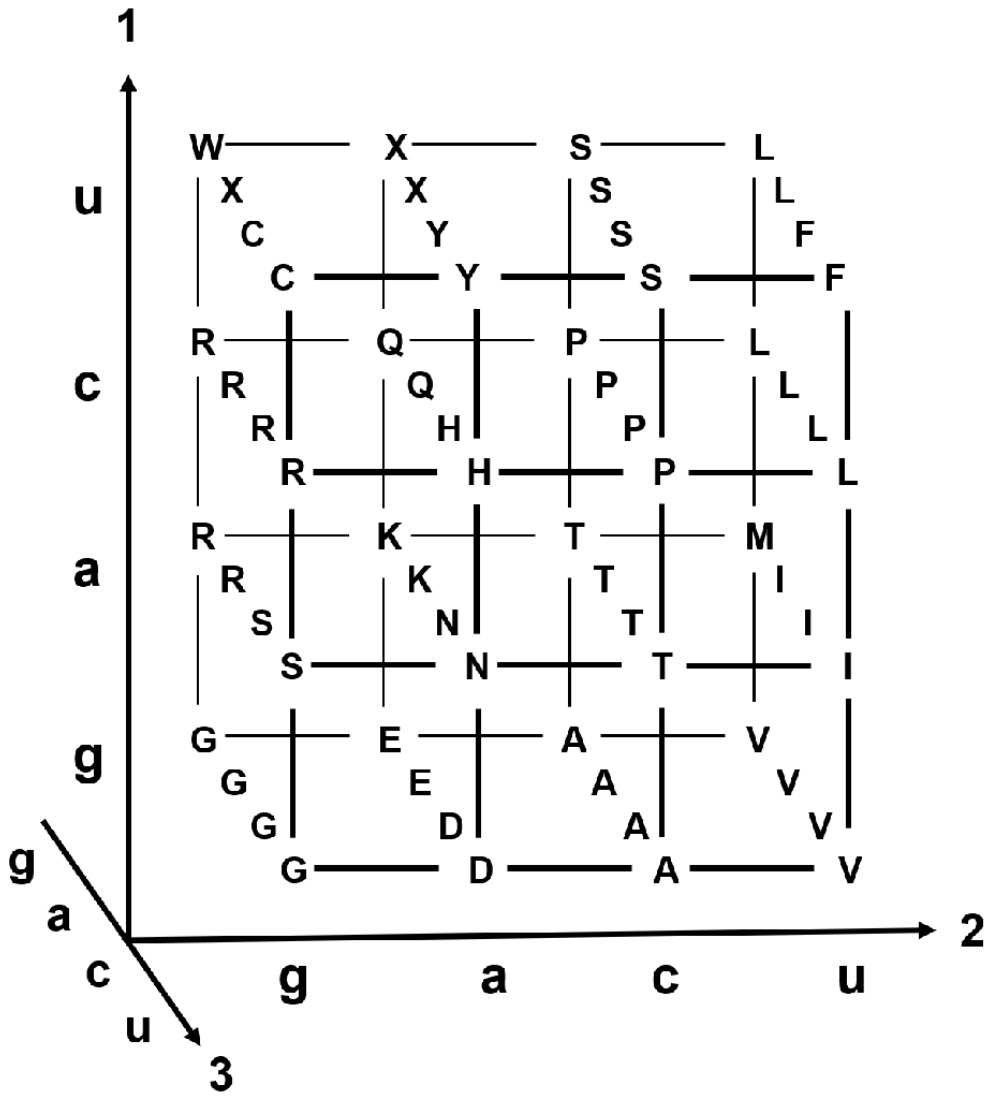
Douglas C. Youvan

doug@youvan.com

September 6, 2023

In a collaboration between a human and artificial intelligence, the authors employ Mathematica to delve into the genetic code's "topological compactness." Using a 64-point 3D grid that spatially represents the 64 RNA codons, the paper assigns each grid point to one of the 20 standard amino acids or one of the three stop codons. With the computational aid of the language model GPT-4 and the mathematical software Mathematica, an adjacency matrix is then generated to capture the nuanced connections between these amino acids. The study evaluates the number of closed loops associated with each amino acid. This method reveals aspects of the geometric and topological layout inherent in the genetic code. A cube arranged as GACU or its mirror UCAG shows the maximization of self-closed loops equal to 44. All other cubes have fewer closed loops. The computer code was generated by GPT-4 in collaboration with the author and follows the 4! = 24 graphs shown herein. This AI-automated Centaur Intelligence confirms the findings of an earlier preprint.

Keywords: Genetic Code, Topology, Graph Theory, Codons, Centaur Intelligence, Artificial Intelligence, AI, GPT-4, Mathematica



Designated Cube 1, the order: g, a, c, u yields a structure with the most closed-loops as shown below. The mirror of this cube (u, c, a, g) is a second solution.

https://www.researchgate.net/publication/361250677_Graph_Analysis_of_the_Genetic_Code_Using_Closed-Loops_Shows_Isotropic_Axes_Using_the_Order_GACU_Yields_Maximum_Topological_Compactness

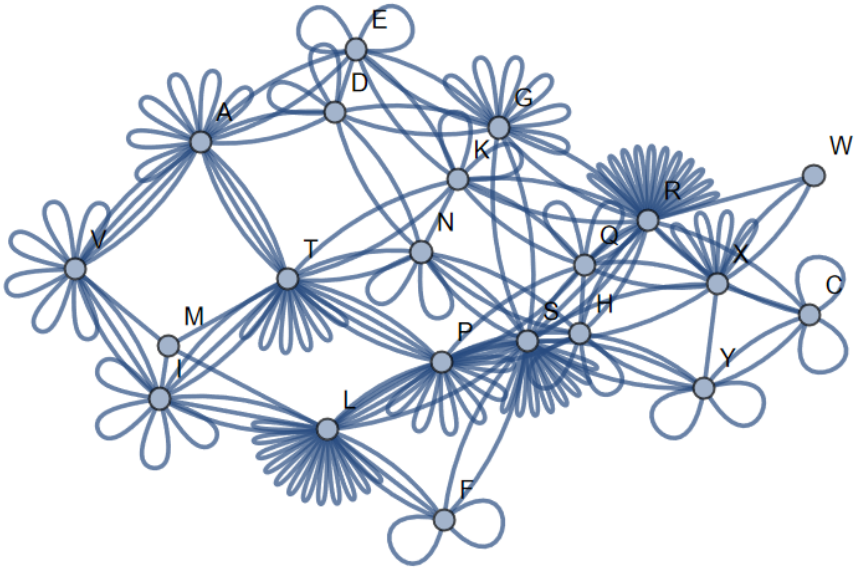
Permutation: {G, A, C, U}

Self-closed loops per amino acid:

⟨|A → 3., C → 1., D → 1., E → 1., F → 1., G → 3., H → 1., I → 2., K → 1., L → 6.,
M → 0, N → 1., P → 3., Q → 1., R → 6., S → 4., T → 3., V → 3., W → 0, X → 2., Y → 1.|⟩

Total number of self-closed loops: 44.

Graph:



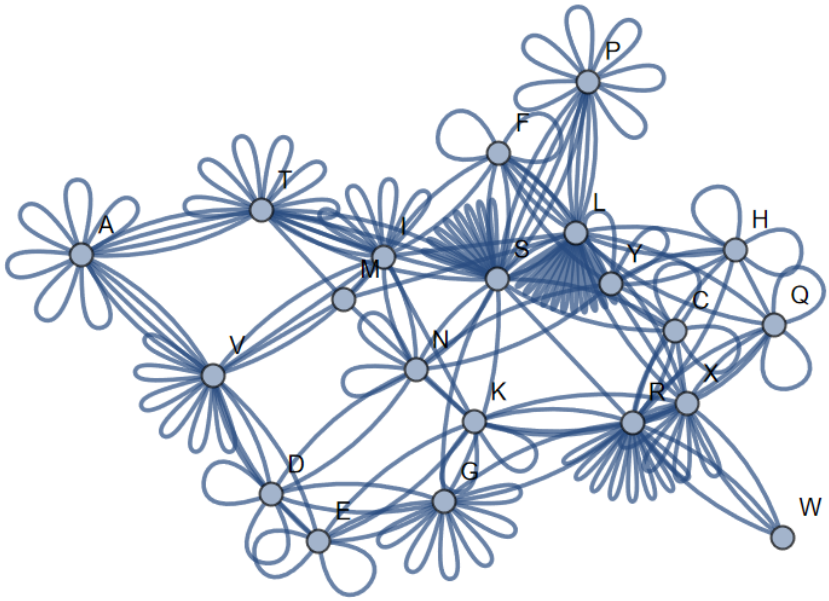
Permutation: {G, A, U, C}

Self-closed loops per amino acid:

⟨|A → 3., C → 1., D → 1., E → 1., F → 1., G → 3., H → 1., I → 2., K → 1., L → 6.,
M → 0, N → 1., P → 3., Q → 1., R → 4., S → 4., T → 3., V → 3., W → 0, X → 2., Y → 1.|⟩

Total number of self-closed loops: 42.

Graph:



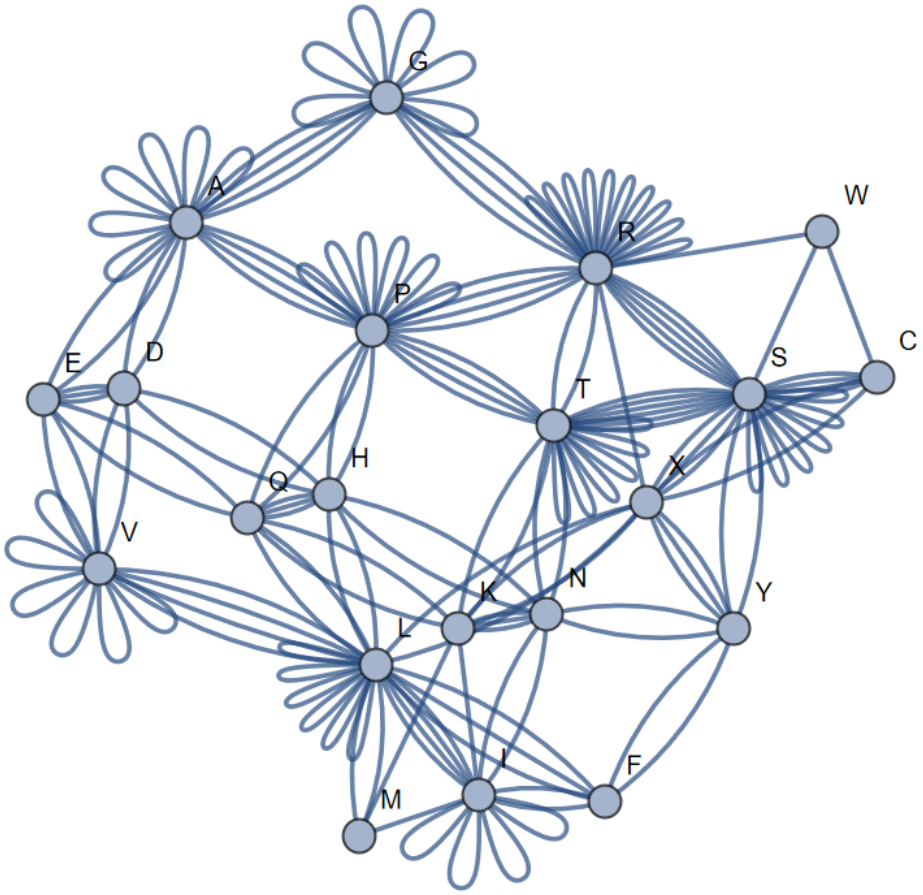
Permutation: {G, C, A, U}

Self-closed loops per amino acid:

⟨|A → 3., C → 0, D → 0, E → 0, F → 0, G → 3., H → 0, I → 2., K → 0, L → 3.,
M → 0, N → 0, P → 3., Q → 0, R → 5., S → 3., T → 3., V → 3., W → 0, X → 0, Y → 0|⟩

Total number of self-closed loops: 28.

Graph:



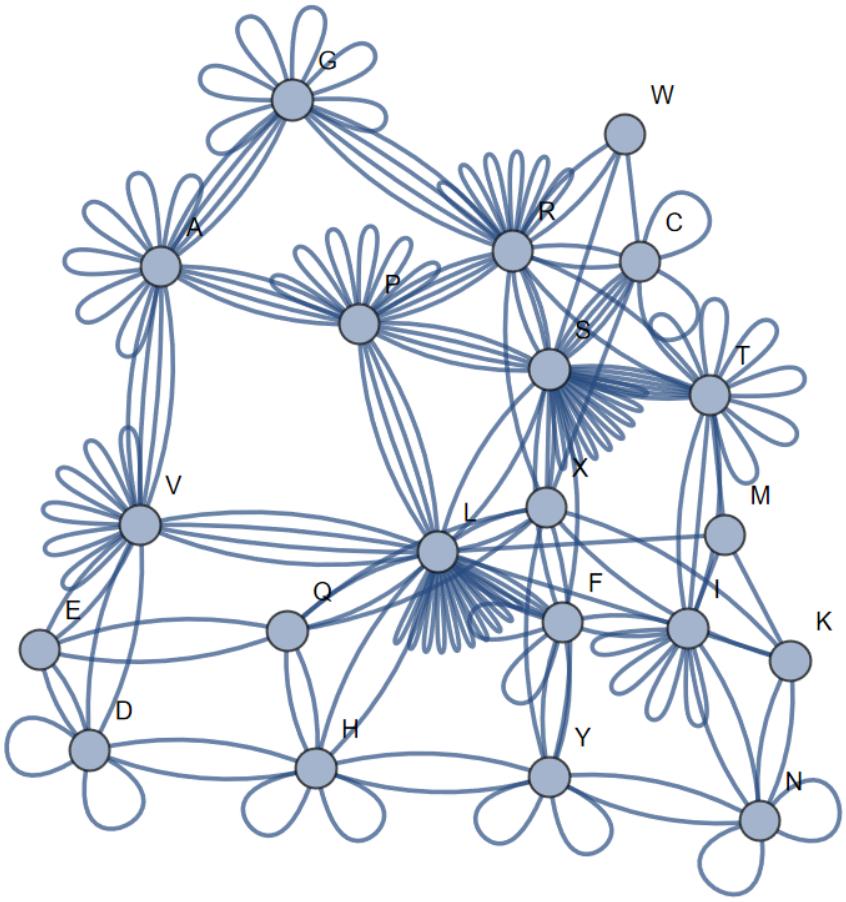
Permutation: {G, C, U, A}

Self-closed loops per amino acid:

⟨|A → 3., C → 1., D → 1., E → 0, F → 1., G → 3., H → 1., I → 2., K → 0, L → 5.,
M → 0, N → 1., P → 3., Q → 0, R → 3., S → 4., T → 3., V → 3., W → 0, X → 0, Y → 1.|⟩

Total number of self-closed loops: 35.

Graph:



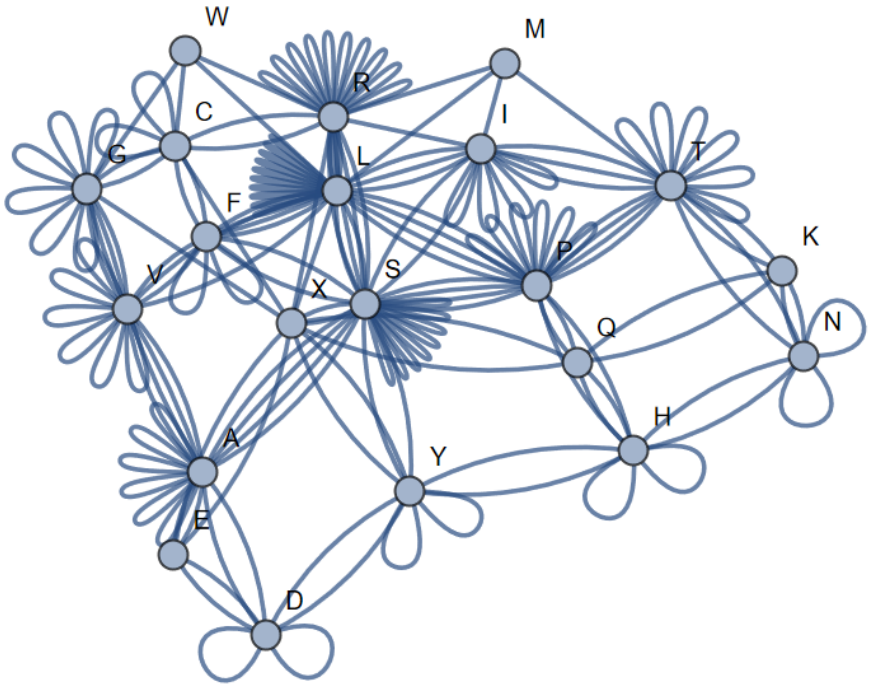
Permutation: {G, U, C, A}

Self-closed loops per amino acid:

⟨|A → 3., C → 1., D → 1., E → 0, F → 1., G → 3., H → 1., I → 2., K → 0, L → 5.,
M → 0, N → 1., P → 3., Q → 0, R → 5., S → 4., T → 3., V → 3., W → 0, X → 0, Y → 1.|⟩

Total number of self-closed loops: 37.

Graph:



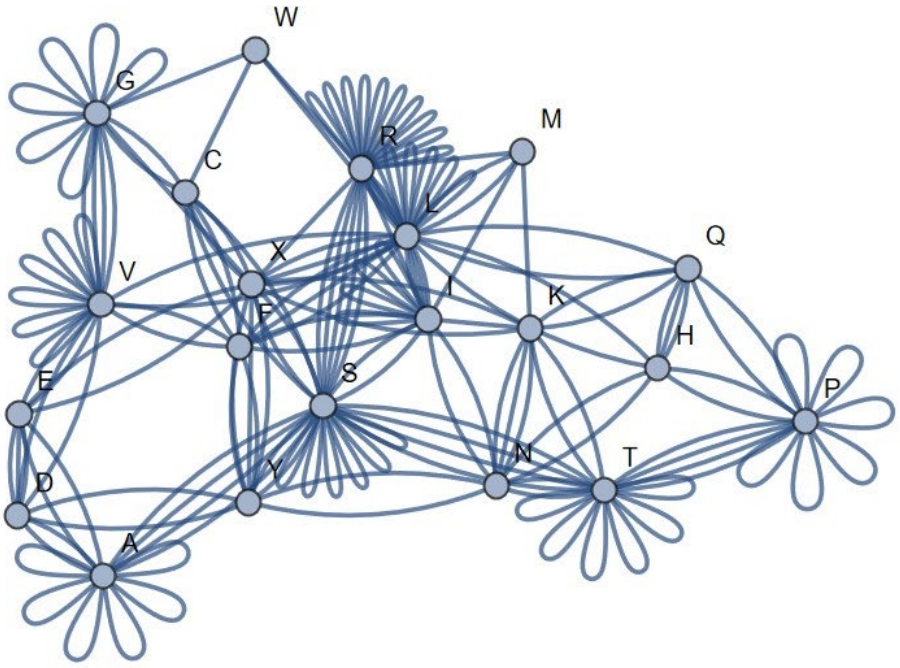
Permutation: {G, U, A, C}

Self-closed loops per amino acid:

⟨|A → 3., C → 0, D → 0, E → 0, F → 0, G → 3., H → 0, I → 2., K → 0, L → 3.,
M → 0, N → 0, P → 3., Q → 0, R → 5., S → 3., T → 3., V → 3., W → 0, X → 0, Y → 0|⟩

Total number of self-closed loops: 28.

Graph:



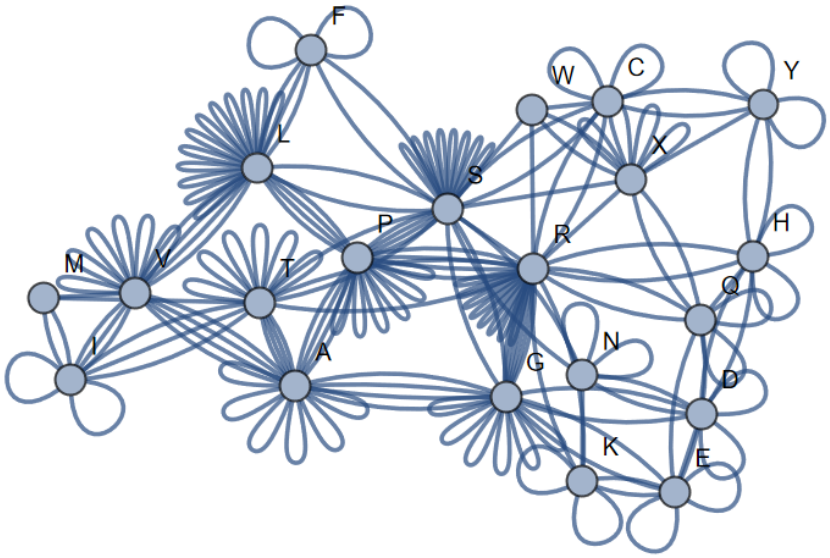
Permutation: {A, G, C, U}

Self-closed loops per amino acid:

⟨|A → 3., C → 1., D → 1., E → 1., F → 1., G → 3., H → 1., I → 1., K → 1., L → 6.,
M → 0, N → 1., P → 3., Q → 1., R → 4., S → 4., T → 3., V → 3., W → 0, X → 2., Y → 1.|⟩

Total number of self-closed loops: 41.

Graph:



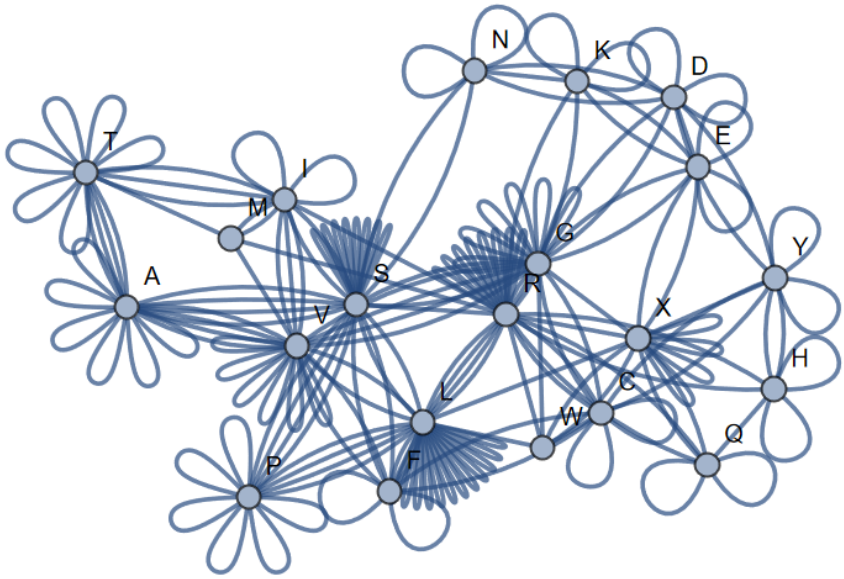
Permutation: {A, G, U, C}

Self-closed loops per amino acid:

⟨|A → 3., C → 1., D → 1., E → 1., F → 1., G → 3., H → 1., I → 1., K → 1., L → 6.,
M → 0, N → 1., P → 3., Q → 1., R → 4., S → 4., T → 3., V → 3., W → 0, X → 2., Y → 1.|⟩

Total number of self-closed loops: 41.

Graph:



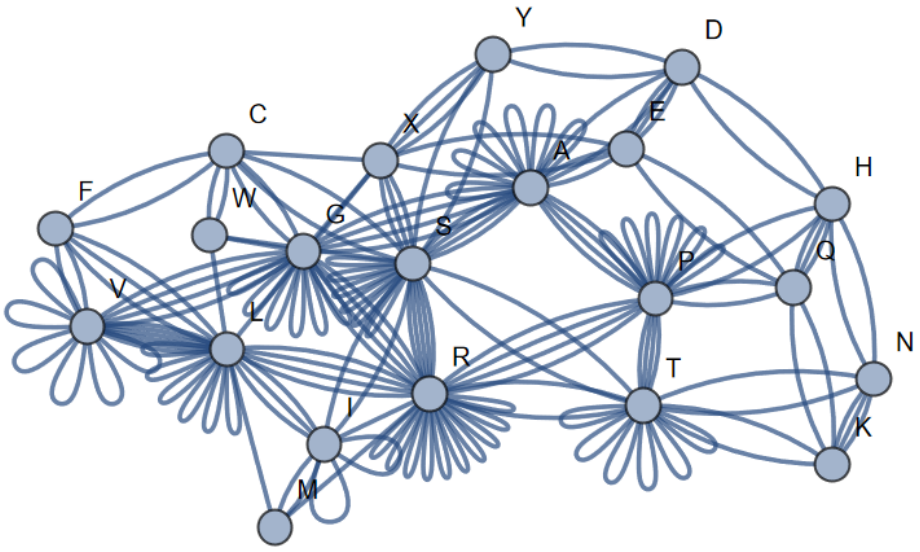
Permutation: {A, C, G, U}

Self-closed loops per amino acid:

⟨|A → 3., C → 0, D → 0, E → 0, F → 0, G → 3., H → 0, I → 1., K → 0, L → 3.,
M → 0, N → 0, P → 3., Q → 0, R → 5., S → 3., T → 3., V → 3., W → 0, X → 0, Y → 0|⟩

Total number of self-closed loops: 27.

Graph:



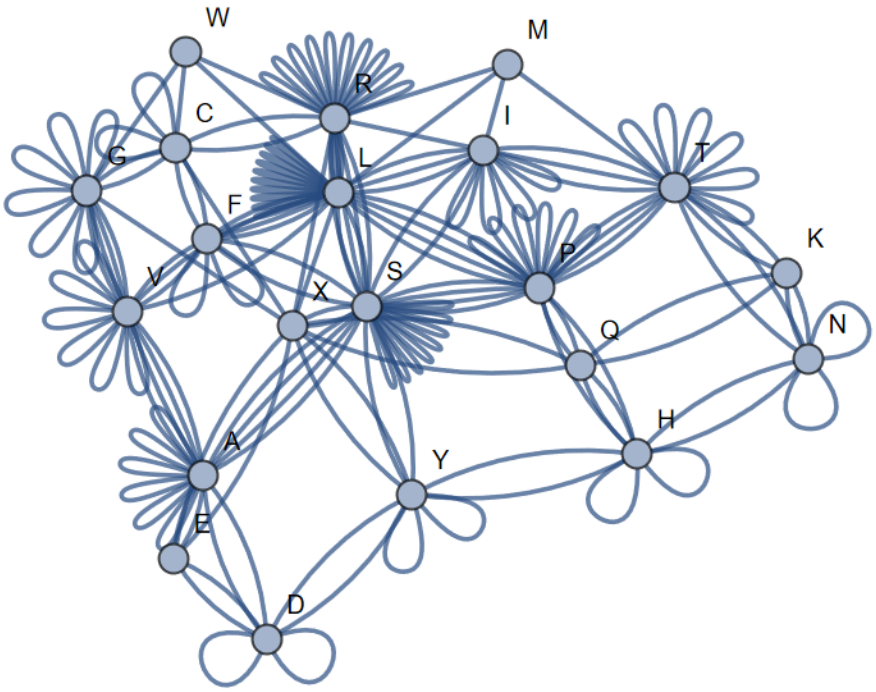
Permutation: {A, C, U, G}

Self-closed loops per amino acid:

⟨|A → 3., C → 1., D → 1., E → 0, F → 1., G → 3., H → 1., I → 2., K → 0, L → 5.,
M → 0, N → 1., P → 3., Q → 0, R → 5., S → 4., T → 3., V → 3., W → 0, X → 0, Y → 1.|⟩

Total number of self-closed loops: 37.

Graph:



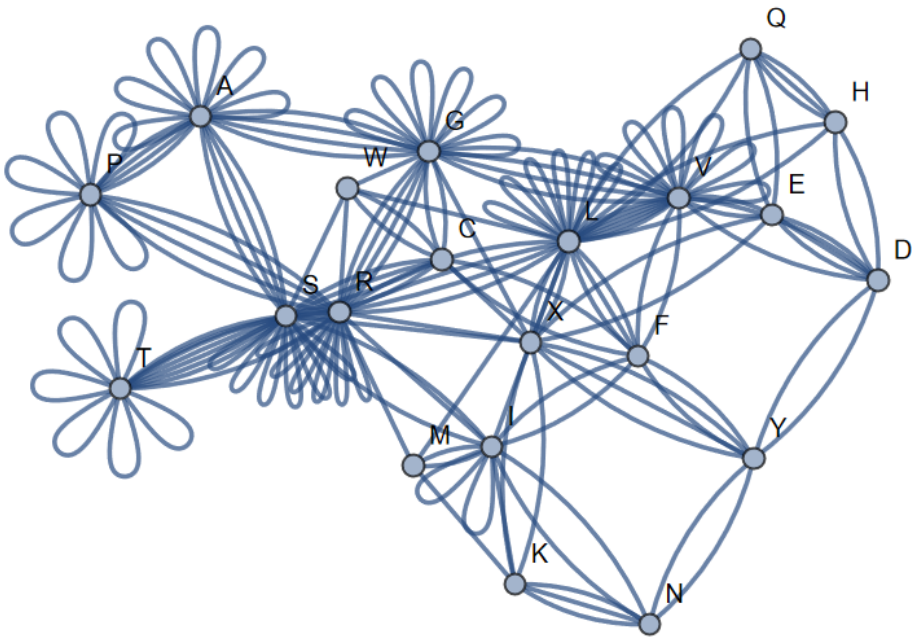
Permutation: {A, U, G, C}

Self-closed loops per amino acid:

⟨|A → 3., C → 0, D → 0, E → 0, F → 0, G → 3., H → 0, I → 1., K → 0, L → 3.,
M → 0, N → 0, P → 3., Q → 0, R → 3., S → 3., T → 3., V → 3., W → 0, X → 0, Y → 0|⟩

Total number of self-closed loops: 25.

Graph:



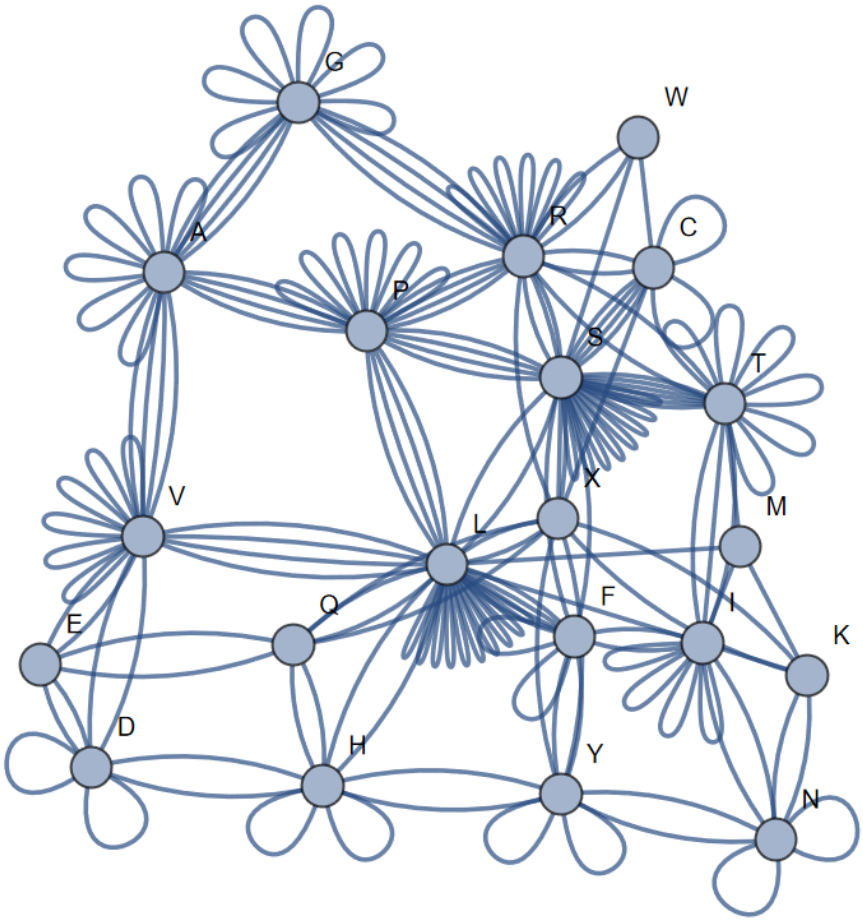
Permutation: {A, U, C, G}

Self-closed loops per amino acid:

⟨|A → 3., C → 1., D → 1., E → 0, F → 1., G → 3., H → 1., I → 2., K → 0, L → 5.,
M → 0, N → 1., P → 3., Q → 0, R → 3., S → 4., T → 3., V → 3., W → 0, X → 0, Y → 1.|⟩

Total number of self-closed loops: 35.

Graph:



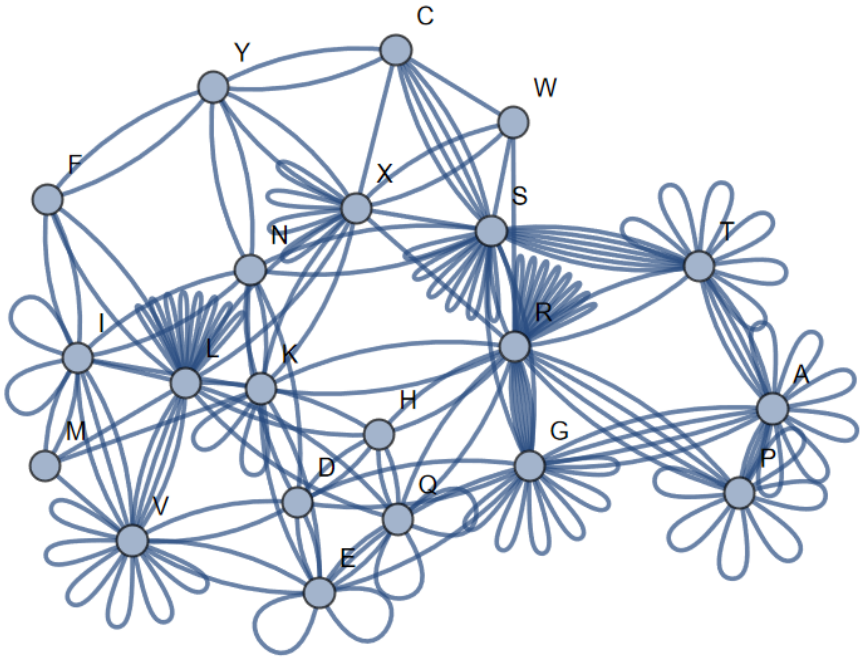
Permutation: {C, G, A, U}

Self-closed loops per amino acid:

⟨|A → 3., C → 0, D → 0, E → 1., F → 0, G → 3., H → 0, I → 1., K → 1., L → 4.,
M → 0, N → 0, P → 3., Q → 1., R → 4., S → 3., T → 3., V → 3., W → 0, X → 2., Y → 0|⟩

Total number of self-closed loops: 32.

Graph:



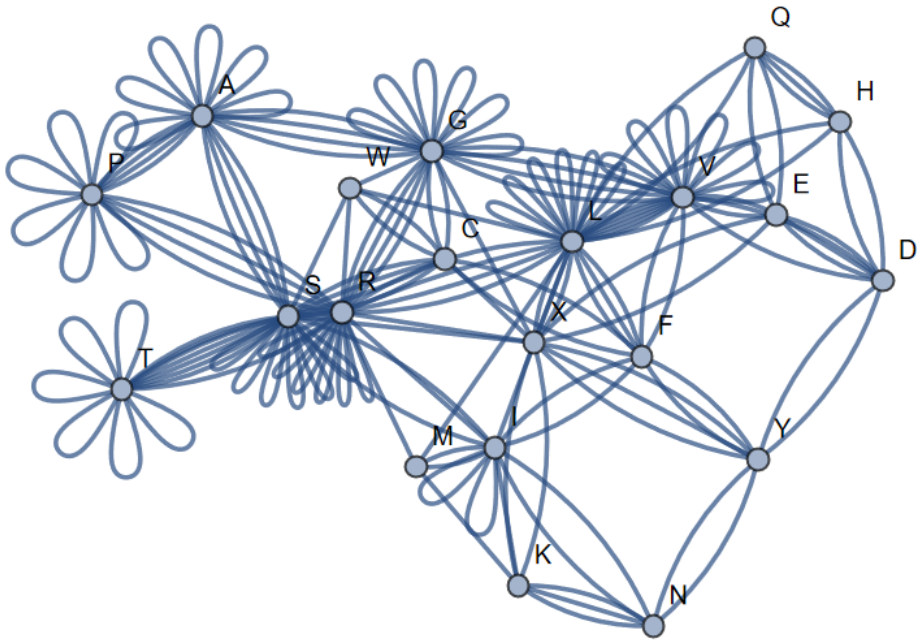
Permutation: {C, G, U, A}

Self-closed loops per amino acid:

⟨|A → 3., C → 0, D → 0, E → 0, F → 0, G → 3., H → 0, I → 1., K → 0, L → 3.,
M → 0, N → 0, P → 3., Q → 0, R → 3., S → 3., T → 3., V → 3., W → 0, X → 0, Y → 0|⟩

Total number of self-closed loops: 25.

Graph:



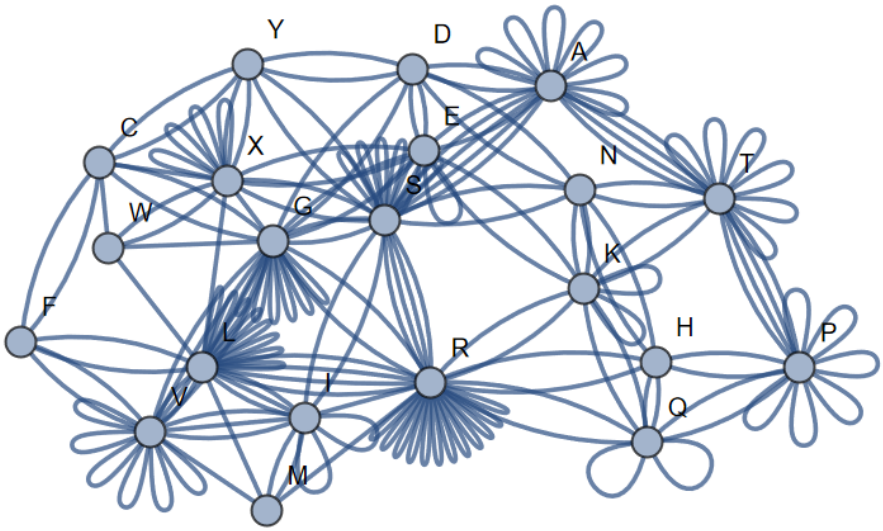
Permutation: {C, A, G, U}

Self-closed loops per amino acid:

⟨|A → 3., C → 0, D → 0, E → 1., F → 0, G → 3., H → 0, I → 1., K → 1., L → 4.,
M → 0, N → 0, P → 3., Q → 1., R → 6., S → 3., T → 3., V → 3., W → 0, X → 2., Y → 0|⟩

Total number of self-closed loops: 34.

Graph:



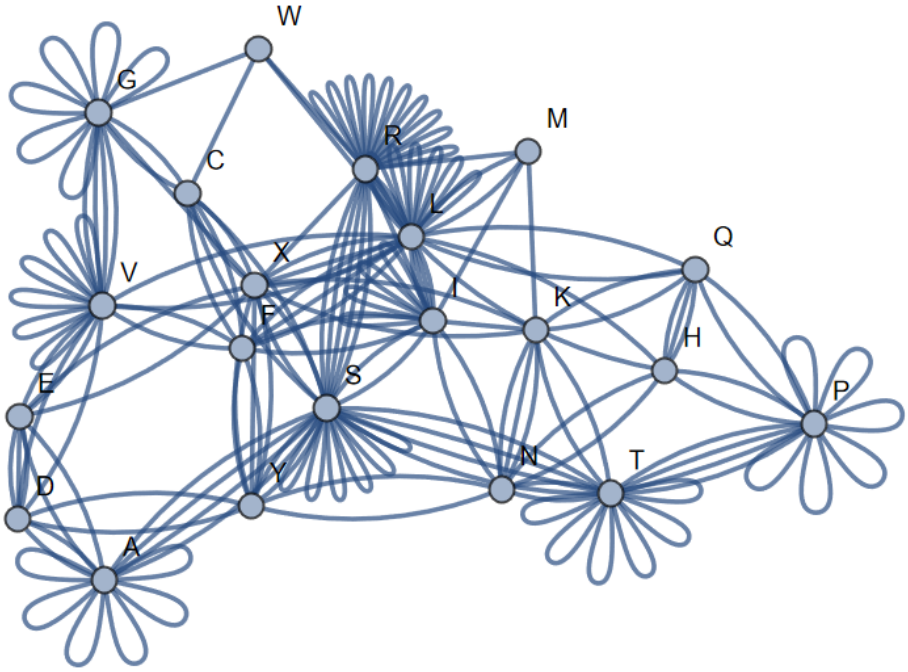
Permutation: {C, A, U, G}

Self-closed loops per amino acid:

⟨|A → 3., C → 0, D → 0, E → 0, F → 0, G → 3., H → 0, I → 2., K → 0, L → 3.,
M → 0, N → 0, P → 3., Q → 0, R → 5., S → 3., T → 3., V → 3., W → 0, X → 0, Y → 0|⟩

Total number of self-closed loops: 28.

Graph:



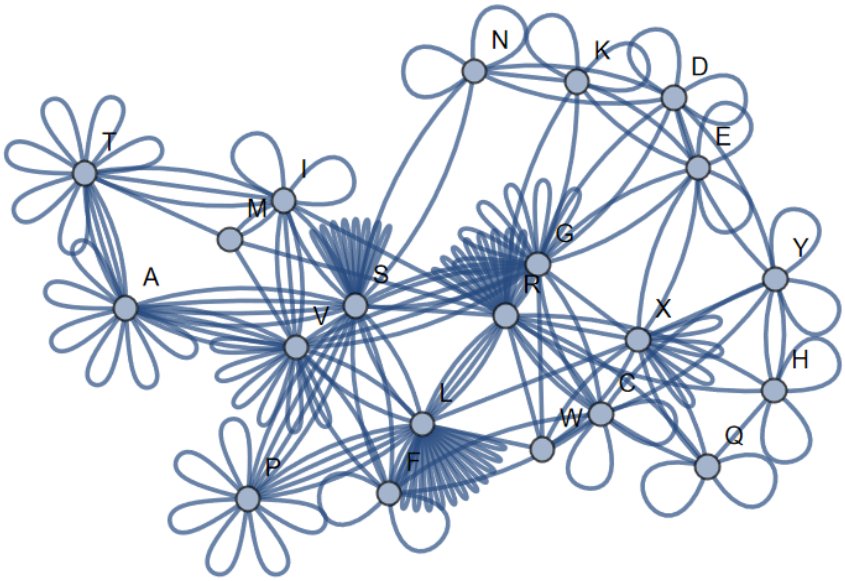
Permutation: {C, U, G, A}

Self-closed loops per amino acid:

⟨|A → 3., C → 1., D → 1., E → 1., F → 1., G → 3., H → 1., I → 1., K → 1., L → 6.,
M → 0, N → 1., P → 3., Q → 1., R → 4., S → 4., T → 3., V → 3., W → 0, X → 2., Y → 1.|⟩

Total number of self-closed loops: 41.

Graph:



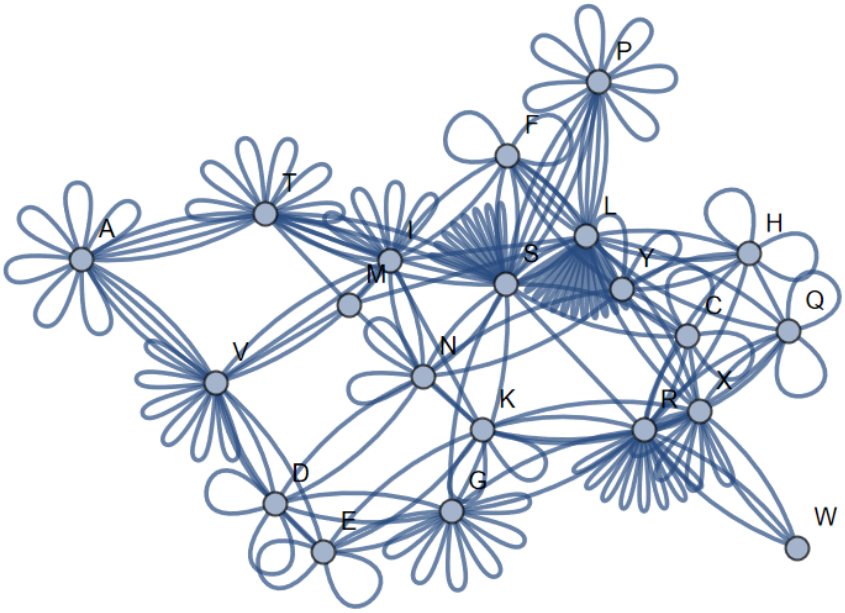
Permutation: {C, U, A, G}

Self-closed loops per amino acid:

⟨|A → 3., C → 1., D → 1., E → 1., F → 1., G → 3., H → 1., I → 2., K → 1., L → 6.,
M → 0, N → 1., P → 3., Q → 1., R → 4., S → 4., T → 3., V → 3., W → 0, X → 2., Y → 1.|⟩

Total number of self-closed loops: 42.

Graph:



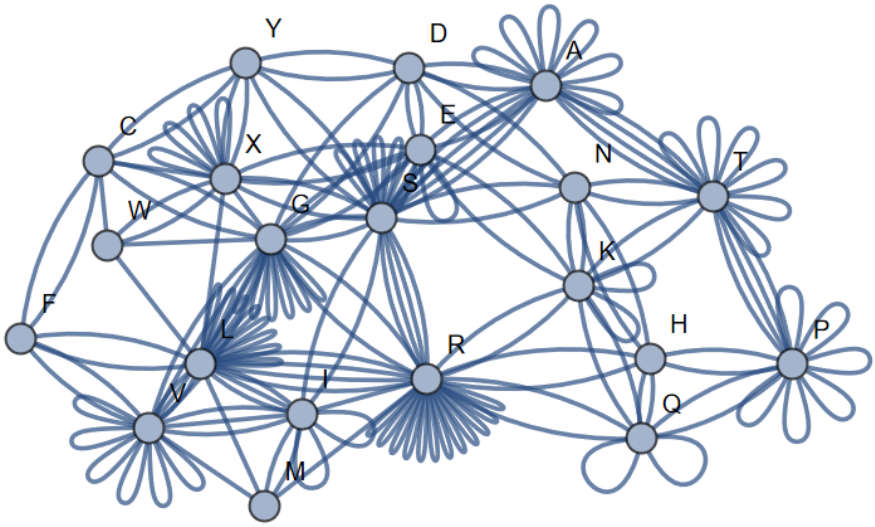
Permutation: {U, G, A, C}

Self-closed loops per amino acid:

⟨|A → 3., C → 0, D → 0, E → 1., F → 0, G → 3., H → 0, I → 1., K → 1., L → 4.,
M → 0, N → 0, P → 3., Q → 1., R → 6., S → 3., T → 3., V → 3., W → 0, X → 2., Y → 0|⟩

Total number of self-closed loops: 34.

Graph:



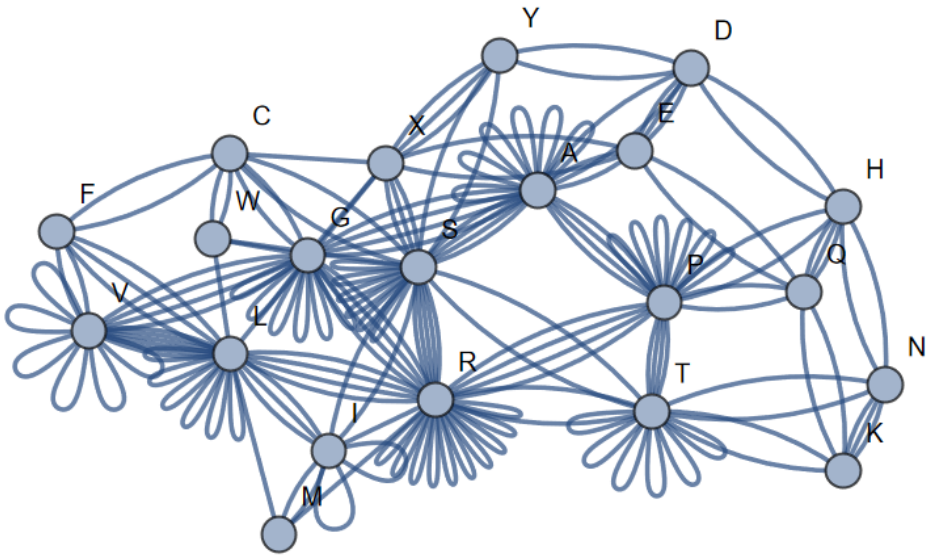
Permutation: {U, G, C, A}

Self-closed loops per amino acid:

⟨|A → 3., C → 0, D → 0, E → 0, F → 0, G → 3., H → 0, I → 1., K → 0, L → 3.,
M → 0, N → 0, P → 3., Q → 0, R → 5., S → 3., T → 3., V → 3., W → 0, X → 0, Y → 0|⟩

Total number of self-closed loops: 27.

Graph:



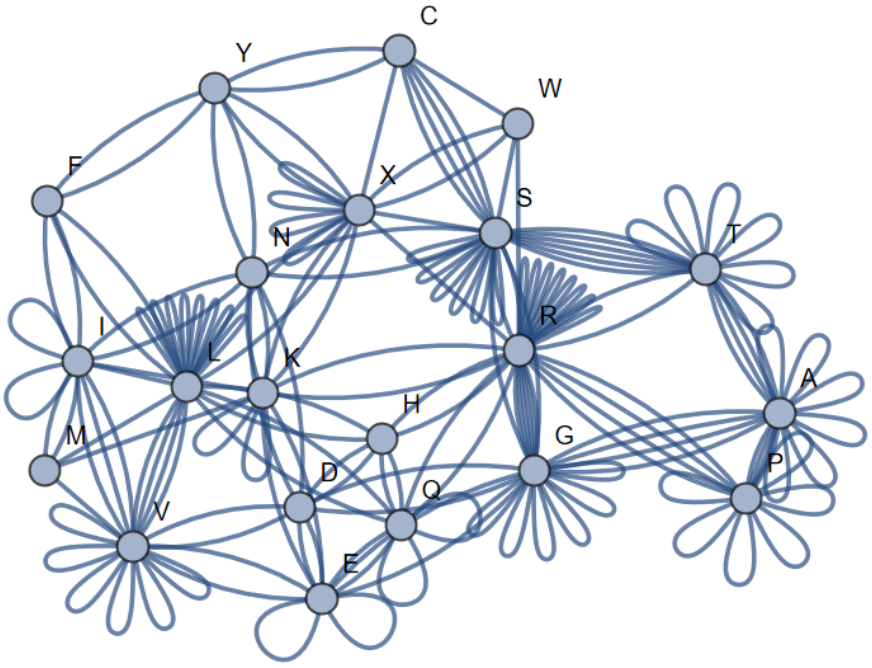
Permutation: {U, A, G, C}

Self-closed loops per amino acid:

⟨|A → 3., C → 0, D → 0, E → 1., F → 0, G → 3., H → 0, I → 1., K → 1., L → 4.,
M → 0, N → 0, P → 3., Q → 1., R → 4., S → 3., T → 3., V → 3., W → 0, X → 2., Y → 0|⟩

Total number of self-closed loops: 32.

Graph:



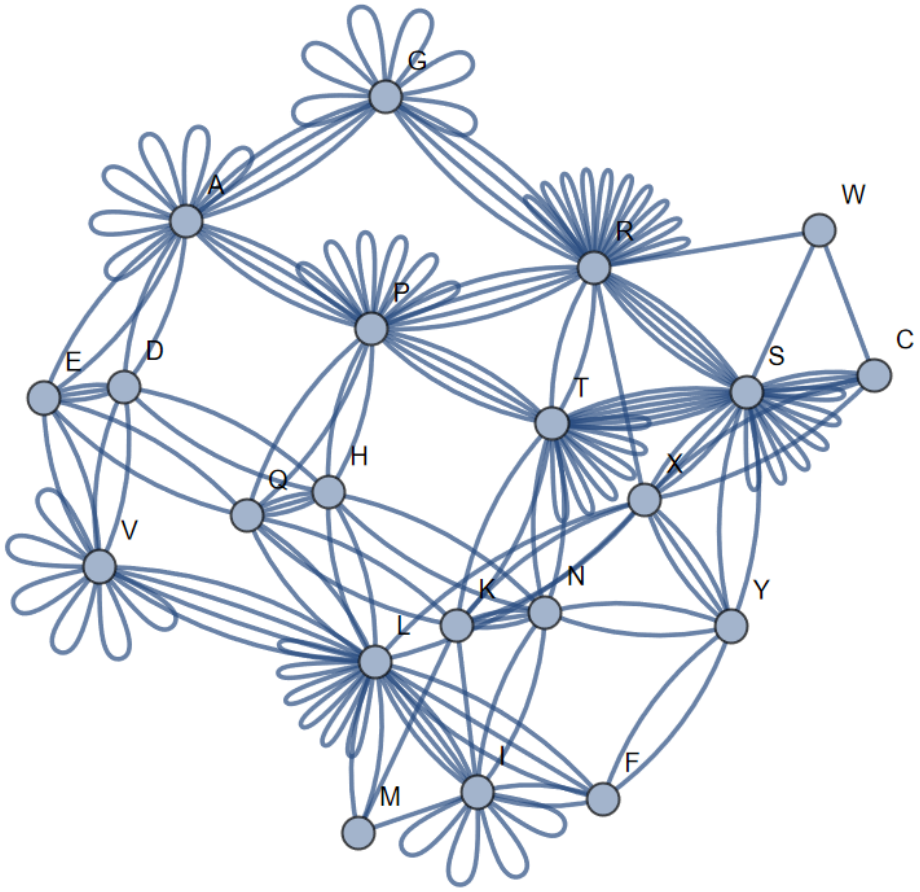
Permutation: {U, A, C, G}

Self-closed loops per amino acid:

⟨|A → 3., C → 0, D → 0, E → 0, F → 0, G → 3., H → 0, I → 2., K → 0, L → 3.,
M → 0, N → 0, P → 3., Q → 0, R → 5., S → 3., T → 3., V → 3., W → 0, X → 0, Y → 0|⟩

Total number of self-closed loops: 28.

Graph:



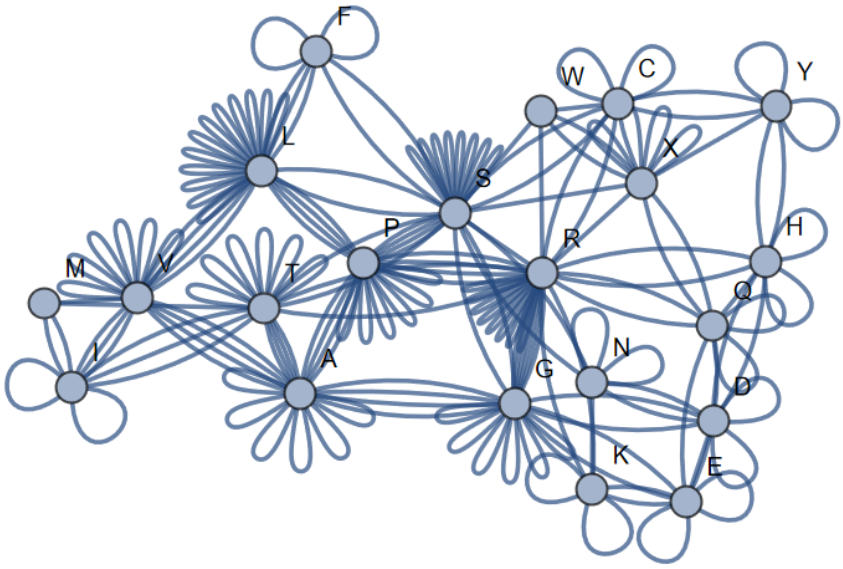
Permutation: {U, C, G, A}

Self-closed loops per amino acid:

⟨|A → 3., C → 1., D → 1., E → 1., F → 1., G → 3., H → 1., I → 1., K → 1., L → 6.,
M → 0, N → 1., P → 3., Q → 1., R → 4., S → 4., T → 3., V → 3., W → 0, X → 2., Y → 1.|⟩

Total number of self-closed loops: 41.

Graph:



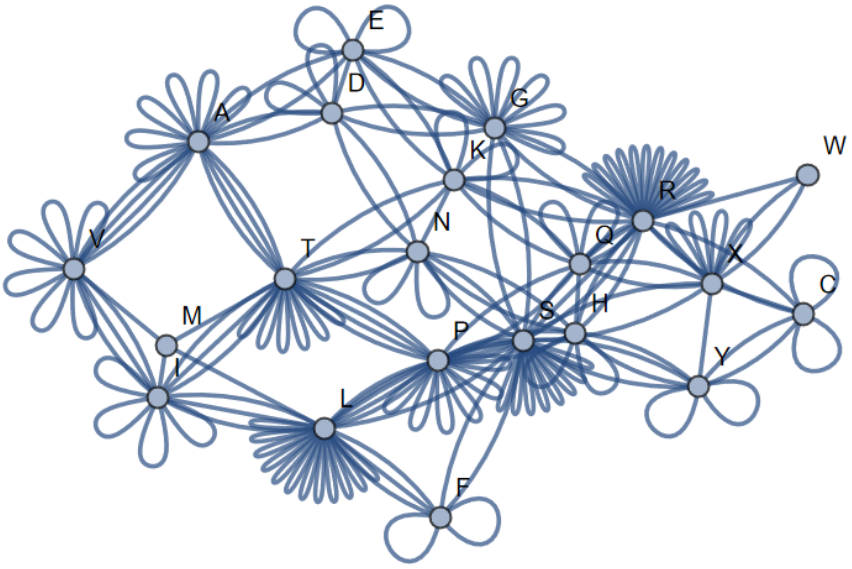
Permutation: {U, C, A, G}

Self-closed loops per amino acid:

⟨|A → 3., C → 1., D → 1., E → 1., F → 1., G → 3., H → 1., I → 2., K → 1., L → 6.,
M → 0, N → 1., P → 3., Q → 1., R → 6., S → 4., T → 3., V → 3., W → 0, X → 2., Y → 1.|⟩

Total number of self-closed loops: 44.

Graph:



```
(*Initialize the list to keep results for each permutation*)results = {};

(*Get all permutations of "G","A","C","U"*)
allPermutations = Permutations[{"G", "A", "C", "U"}];

(*Loop over all permutations*)
For[p = 1, p ≤ Length[allPermutations], p++, perm = allPermutations[[p]];
  (*Define the genetic code based on the current permutation*)
  geneticCode = <|"UUU" → "F", "UCU" → "S", "UAU" → "Y", "UGU" → "C", "UUC" → "F", "UCC" → "S", "UAC" → "Y", "UGC" → "C", "UUA" → "L", "UCA" → "S", "UAA" → "X",
    "UGA" → "X", "UUG" → "L", "UCG" → "S", "UAG" → "X", "UGG" → "W", "CUU" → "L", "CCU" → "P", "CAU" → "H", "CGU" → "R", "CUC" → "L", "CCC" → "P", "CAC" → "H",
    "CGC" → "R", "CUA" → "L", "CCA" → "P", "CAA" → "Q", "CGA" → "R", "CUG" → "L", "CCG" → "P", "CAG" → "Q", "CGG" → "R", "AUU" → "I", "ACU" → "T", "AAU" → "N",
    "AGU" → "S", "AUC" → "I", "ACC" → "T", "AAC" → "N", "AGC" → "S", "AUA" → "I", "ACA" → "T", "AAA" → "K", "AGA" → "R", "AUG" → "M", "ACG" → "T", "AAG" → "K",
    "AGG" → "R", "GUU" → "V", "GCU" → "A", "GAU" → "D", "GGU" → "G", "GUC" → "V", "GCC" → "A", "GAC" → "D", "GGC" → "G", "GUA" → "V", "GCA" → "A", "GAA" → "E",
    "GGA" → "G", "GUG" → "V", "GCG" → "A", "GAG" → "E", "GGG" → "G"|>;
  (*Create the 3D lattice*)lattice = Table[geneticCode[StringJoin @@ {x, y, z}], {x, perm}, {y, perm}, {z, perm}];
  (*Initialize adjacency matrix and self-closed loops counter*)aminoAcids = Union[Values[geneticCode]];
  adjacencyMatrix = ConstantArray[0, {Length[aminoAcids], Length[aminoAcids]}];
  selfClosedLoops = AssociationThread[aminoAcids → 0];
  (*Helper function to get 3D neighbors*)
  getNeighbors[i_, j_, k_] := Select[{{i - 1, j, k}, {i + 1, j, k}, {i, j - 1, k}, {i, j + 1, k}, {i, j, k - 1}, {i, j, k + 1}}, AllTrue[#, 1 ≤ # ≤ 4 &] &];
  (*Populate adjacency matrix and count self-closed loops*)For[i = 1, i ≤ 4, i++, For[j = 1, j ≤ 4, j++, For[k = 1, k ≤ 4, k++, src = lattice[[i, j, k]];
     srcIndex = FirstPosition[aminoAcids, src][[1]];
     neighbors = getNeighbors[i, j, k];
     For[n = 1, n ≤ Length[neighbors], n++, neighbor = neighbors[[n]];
      dest = lattice[[Sequence @@ neighbor]];
      destIndex = FirstPosition[aminoAcids, dest][[1]];
adjacencyMatrix[[srcIndex, destIndex]] += 1;
If[src == dest, selfClosedLoops[src] += 0.5];
]
]
]
];

(*Create the graph*)
  g = AdjacencyGraph[aminoAcids, adjacencyMatrix, VertexLabels → "Name", VertexSize → Large, EdgeStyle → Thickness[.005]];

(*Show self-closed loops and graph*)
  Print["Permutation: ", perm];
Print["Self-closed loops per amino acid:"];
Print[selfClosedLoops];
Print["Total number of self-closed loops: ", Total[Values[selfClosedLoops]]];
Print["Graph:"];
Print[g];

(*Append the results to the list*)
  AppendTo[results, {"Permutation" → perm, "SelfClosedLoops" → selfClosedLoops, "TotalLoops" → Total[Values[selfClosedLoops]], "Graph" → g}];
];

(*Show all the results*)
Print["All Results:"];
Print[results];
```